BeeGFSファイルシステムを導入します

BeeGFS on NetApp with E-Series Storage

NetApp
January 27, 2026

目次

BeeGFSファイルシステムを導入します

Ansible Playbookの概要

Ansibleを使用したBeeGFS HAクラスタの導入と管理

概要

前のセクションでは、BeeGFS HAクラスタを表すAnsibleインベントリを構築するために必要な手順を説明しました。このセクションでは、ネットアップがクラスタの導入と管理を行うAnsibleによる自動化を紹介します。

Ansible：重要な概念

開始する前に、Ansibleの主要な概念を理解しておくと役立ちます。

- Ansibleインベントリに対して実行されるタスクは、* Playbook *と呼ばれるもので定義されています。
  - Ansibleのほとんどのタスクは*べき等値*となるように設計されているため、何度も実行して、必要な構成や状態が適用されていることを確認することができます。その際、作業を中断したり、不要な更新を加える必要はありません。
- Ansibleで実行される最小単位は*モジュール*です。
  - 一般的なプレイブックでは、複数のモジュールを使用
    - 例：パッケージのダウンロード、構成ファイルの更新、サービスの開始/有効化
  - NetApp Eシリーズシステムを自動化するために、モジュールを配布
- 複雑な自動化はロールとしてより適切にパッケージ化されています。
  - 基本的には、再利用可能なプレイブックを配布するための標準形式です。
  - LinuxホストとBeeGFSファイルシステムに役割を配布します。

AnsibleのBeeGFS HAロール：主な概念

ネットアップ上のBeeGFSの各バージョンの導入と管理に必要なすべての自動化機能がAnsibleのロールとしてパッケージ化され、の一部として提供されます "BeeGFSに対応したNetApp EシリーズAnsibleコレクション"：

- この役割は、BeeGFS用の*インストーラ*と最新の*導入/管理*エンジンの間にあると考えることができます。
  - コードの手法や理念として最新のインフラを活用し、あらゆる規模のストレージインフラをシンプルに管理できます。
  - この"久保スプレー"プロジェクトでは、スケールアウトコンピューティングインフラ向けにKubernetesディストリビューション全体を導入/保守できるようになります。
- この役割は、ネットアップのソリューションでBeeGFSをパッケージ化、配布、保守するためにネットアップが使用する*ソフトウェア定義*形式です。
  - Linuxディストリビューション全体や大きなイメージを配布することなく、「アプライアンスのような」エクスペリエンスを実現できるように努力してください。

- カスタムのBeeGFSターゲットとIPアドレスに対応したネットアップがオーサリングしたOpen Cluster Framework（OCF）準拠のクラスタリソースエージェントで構成され、高度なPacemakerとBeeGFSを統合するための監視機能が提供されます。

- この役割は、単に導入を「自動化」するものではなく、以下を含むファイルシステムのライフサイクル全体を管理することを目的としています。
  - サービス単位またはクラスタ全体の設定変更および更新を適用する。
  - ハードウェアの問題が解決されたあとのクラスタの修復とリカバリの自動化
  - BeeGFSとネットアップのボリュームを使用した広範なテストに基づいてデフォルト値を設定することで、パフォーマンスの調整を簡易化
  - 構成のずれの検証と修正

ネットアップは、向けのAnsibleのロールも提供しています "BeeGFSクライアント"必要に応じて、BeeGFSのインストールとファイルシステムのマウントを行い、/GPU/ログインノードを計算します。

BeeGFS HAクラスタを導入します

プレイブックを使用してBeeGFS HAクラスタを導入するために実行するタスクを指定します。

概要

このセクションでは、ネットアップでBeeGFSを導入/管理するために使用する標準的なプレイブックを組み立てる方法について説明します。

手順

Ansible Playbookを作成

ファイルを作成します `playbook.yml` 次のように入力します。

1. 最初に、一連のタスクを定義します（一般的には、と呼ばれます） "再生"）が実行されるのはNetApp Eシリーズのブロックノードだけです。インストールを実行する前に確認を求めて（誤ってプレイブックが実行されないように）、をインポートします `nar_santricity_management` ロール。このロールは、で定義されている一般的なシステム構成の適用を処理します `group_vars/eseries_storage_systems.yml` または個人 `host_vars/<BLOCK NODE>.yml` ファイル。

```
- hosts: eseries_storage_systems
  gather_facts: false
  collections:
    - netapp_eseries.santricity
  tasks:
    - name: Verify before proceeding.
      pause:
        prompt: "Are you ready to proceed with running the BeeGFS HA
role? Depending on the size of the deployment and network performance
between the Ansible control node and BeeGFS file and block nodes this
can take awhile (10+ minutes) to complete."
    - name: Configure NetApp E-Series block nodes.
      import_role:
        name: nar_santricity_management
```

2. すべてのファイルノードおよびブロックノードに対して実行する再生を定義します。

```
- hosts: all
  any_errors_fatal: true
  gather_facts: false
  collections:
    - netapp_eseries.beegfs
```

3. このアプローチでは、必要に応じて、HAクラスタを導入する前に実行する一連の「事前タスク」を定義できます。これは、Pythonなどの前提条件を確認してインストールするのに役立ちます。また、提供されたAnsibleタグがサポートされていることを確認するなど、任意のプリフライトチェックを実行することもできます。

```
  pre_tasks:
    - name: Ensure a supported version of Python is available on all
file nodes.
      block:
        - name: Check if python is installed.
          failed_when: false
          changed_when: false
          raw: python --version
          register: python_version

        - name: Check if python3 is installed.
          raw: python3 --version
          failed_when: false
          changed_when: false
          register: python3_version
          when: 'python_version["rc"] != 0 or (python_version["stdout"]
```

```
| regex_replace("Python ", "")) is not version("3.0", ">=")'

        - name: Install python3 if needed.
          raw: |
            id=$(grep "^ID=" /etc/*release* | cut -d= -f 2 | tr -d '"')
            case $id in
              ubuntu) sudo apt install python3 ;;
              rhel|centos) sudo yum -y install python3 ;;
              sles) sudo zypper install python3 ;;
            esac
          args:
            executable: /bin/bash
          register: python3_install
          when: python_version['rc'] != 0 and python3_version['rc'] != 0
          become: true

        - name: Create a symbolic link to python from python3.
          raw: ln -s /usr/bin/python3 /usr/bin/python
          become: true
          when: python_version['rc'] != 0
      when: inventory_hostname not in
groups[beegfs_ha_ansible_storage_group]

    - name: Verify any provided tags are supported.
      fail:
        msg: "{{ item }} tag is not a supported BeeGFS HA tag. Rerun
your playbook command with --list-tags to see all valid playbook tags."
      when: 'item not in ["all", "storage", "beegfs_ha",
"beegfs_ha_package", "beegfs_ha_configure",
"beegfs_ha_configure_resource", "beegfs_ha_performance_tuning",
"beegfs_ha_backup", "beegfs_ha_client"]'
      loop: "{{ ansible_run_tags }}"
```

4. 最後に、導入するBeeGFSのバージョンに応じてBeeGFS HAロールをインポートします。

```
  tasks:
    - name: Verify the BeeGFS HA cluster is properly deployed.
      import_role:
        name: beegfs_ha_7_4 # Alternatively specify: beegfs_ha_7_3.
```

BeeGFS HAロールは、サポートされるメジャーマイナーバージョンのBeeGFSごとに維持されます。これにより、ユーザはメジャー/マイナーバージョンをいつアップグレードするかを選択できます。現在、BeeGFS 7.3.x(`beegfs_7_3`）またはBeeGFS 7.2.x(`beegfs_7_2`）のいずれかがサポートされています。デフォルトでは、どちらのロールでも最新のBeeGFSパッチバージョンがリリース時に導入されますが、必要に応じてこれを上書きして最新のパッチを導入することもできます。"アップグレードガイド"詳細については、最新のを参照してください。

5. オプション：追加のタスクを定義する場合は、タスクの指示を考慮してください `all` ホスト（Eシリーズストレージシステムを含む）またはファイルノードのみ。必要に応じて、を使用して、ファイルノードを対象とした新しいプレイを定義します `- hosts: ha_cluster`。

をクリックします "こちらをご覧ください" に、完全なPlaybookファイルの例を示します。

NetApp Ansibleコレクションをインストールします

AnsibleのBeeGFSコレクションとすべての依存関係は維持されます "Ansible Galaxy"。Ansibleコントロールノードで次のコマンドを実行して最新バージョンをインストールします。

```
ansible-galaxy collection install netapp_eseries.beegfs
```

通常は推奨されませんが、コレクションの特定のバージョンをインストールすることもできます。

```
ansible-galaxy collection install netapp_eseries.beegfs:
==<MAJOR>.<MINOR>.<PATCH>
```

Playbookを実行してください

を含むAnsibleコントロールノードのディレクトリから `inventory.yml` および `playbook.yml` ファイルでは、次のようにプレイブックを実行します。

```
ansible-playbook -i inventory.yml playbook.yml
```

クラスタのサイズによっては、初期導入に20分以上かかることがあります。何らかの理由で導入が失敗した場合は、問題を修正し（ケーブルの接続ミス、ノードの起動など）、Ansibleプレイブックを再起動するだけです。

を指定するときに"共通ファイルノード構成"、接続ベースの認証をAnsibleで自動的に管理するデフォルトオプションを選択した場合、 `connAuthFile`共有シークレットとして使用されているが `<playbook_dir>/files/beegfs/<sysMgmtdHost>_connAuthFile`（デフォルト）に表示されるようになりました。ファイルシステムにアクセスする必要があるクライアントは、この共有シークレットを使用する必要があります。これは、クライアントがを使用して設定されている場合に自動的に処理され"BeeGFSクライアントの役割"ます。

BeeGFSクライアントを導入します

また、Ansibleを使用してBeeGFSクライアントを設定し、ファイルシステムをマウントすることもできます。

概要

BeeGFSファイルシステムにアクセスするには、ファイルシステムをマウントする必要のある各ノードにBeeGFSクライアントをインストールして設定する必要があります。このセクションでは、使用可能なを使用してこれらのタスクを実行する方法について説明します "Ansibleのロール"。

手順

クライアントインベントリファイルを作成します

1. 必要に応じて、Ansibleコントロールノードから、BeeGFSクライアントとして設定する各ホストにパスワードなしのSSHを設定します。

```
ssh-copy-id <user>@<HOSTNAME_OR_IP>
```

2. の下 `host_vars/` をクリックし、という名前のBeeGFSクライアントごとにファイルを作成します `<HOSTNAME>.yml` 次の内容を使用して、プレースホルダテキストに環境に適した情報を入力します。

```
# BeeGFS Client
ansible_host: <MANAGEMENT_IP>
```

3. NetApp Eシリーズホストコレクションのロールを使用して、クライアントがBeeGFSファイルノードに接続するためのInfiniBandインターフェイスまたはイーサネットインターフェイスを設定する場合は、オプションで次のいずれかを指定します。

   a. ネットワークタイプがの場合 "InfiniBand（IPoIBを使用）"：

```
eseries_ipoib_interfaces:
- name: <INTERFACE>  # Example: ib0 or i1b
  address: <IP/SUBNET> # Example: 100.127.100.1/16
- name: <INTERFACE>  # Additional interfaces as needed.
  address: <IP/SUBNET>
```

   b. ネットワークタイプがの場合 "RDMA over Converged Ethernet（RoCE）"：

```
eseries_roce_interfaces:
- name: <INTERFACE>  # Example: eth0.
  address: <IP/SUBNET> # Example: 100.127.100.1/16
- name: <INTERFACE>  # Additional interfaces as needed.
  address: <IP/SUBNET>
```

c. ネットワークタイプがの場合 "イーサネット（TCPのみ、RDMAなし）"：

```
eseries_ip_interfaces:
- name: <INTERFACE>  # Example: eth0.
  address: <IP/SUBNET> # Example: 100.127.100.1/16
- name: <INTERFACE>  # Additional interfaces as needed.
  address: <IP/SUBNET>
```

4. 新しいファイルを作成します `client_inventory.yml` さらに、Ansibleが各クライアントに接続するために使用するユーザを指定します。また、パスワードがAnsibleで権限の昇格（これにはが必要です `ansible_ssh_user` rootにするか、sudo権限を持っているか）：

```
# BeeGFS client inventory.
all:
  vars:
    ansible_ssh_user: <USER>
    ansible_become_password: <PASSWORD>
```

パスワードをプレーンテキストで保存しないでください。代わりにAnsible Vaultを使用します（を参照してください） "Ansibleのドキュメント" Ansible Vaultを使用してコンテンツを暗号化する場合）またはを使用します `--ask-become-pass` プレイブックを実行する際のオプション。

5. を参照してください `client_inventory.yml` ファイルに、の下でBeeGFSクライアントとして設定する必要があるすべてのホストをリストします `beegfs_clients` グループ化し、インラインコメントを参照して、BeeGFSクライアントカーネルモジュールをシステムに構築するために必要な追加設定のコメントを外します。

```
children:
    # Ansible group representing all BeeGFS clients:
    beegfs_clients:
      hosts:
        <CLIENT HOSTNAME>:
        # Additional clients as needed.

      vars:
        # OPTION 1: If you're using the NVIDIA OFED drivers and they are
already installed:
        #eseries_ib_skip: True # Skip installing inbox drivers when
using the IPoIB role.
        #beegfs_client_ofed_enable: True
        #beegfs_client_ofed_include_path:
"/usr/src/ofa_kernel/default/include"

        # OPTION 2: If you're using inbox IB/RDMA drivers and they are
already installed:
        #eseries_ib_skip: True # Skip installing inbox drivers when
using the IPoIB role.

        # OPTION 3: If you want to use inbox IB/RDMA drivers and need
them installed/configured.
        #eseries_ib_skip: False # Default value.
        #beegfs_client_ofed_enable: False # Default value.
```

NVIDIA OFEDドライバを使用する場合は、beegfs_client_ofed_include_pathが、使用しているLinuxのインストールに適した「ヘッダーインクルードパス」を指定していることを確認してください。詳細については、BeeGFSのドキュメントを参照してください "RDMAのサポート"。

6. を参照してください `client_inventory.yml` ファイルで、以前に定義した任意の下にマウントするBeeGFSファイルシステムを一覧表示します `vars`：

```
        beegfs_client_mounts:
          - sysMgmtdHost: <IP ADDRESS>  # Primary IP of the BeeGFS
management service.
            mount_point: /mnt/beegfs    # Path to mount BeeGFS on the
client.
            connInterfaces:
              - <INTERFACE> # Example: ibs4f1
              - <INTERFACE>
            beegfs_client_config:
              # Maximum number of simultaneous connections to the same
node.
              connMaxInternodeNum: 128 # BeeGFS Client Default: 12
              # Allocates the number of buffers for transferring IO.
              connRDMABufNum: 36 # BeeGFS Client Default: 70
              # Size of each allocated RDMA buffer
              connRDMABufSize: 65536 # BeeGFS Client Default: 8192
              # Required when using the BeeGFS client with the shared-
disk HA solution.
              # This does require BeeGFS targets be mounted in the
default "sync" mode.
              # See the documentation included with the BeeGFS client
role for full details.
              sysSessionChecksEnabled: false
        # Specify additional file system mounts for this or other file
systems.
```

7. BeeGFS 7.2.7および7.3.1以降で"接続認証"は、設定または明示的に無効にする必要があります。を指定するときに接続ベースの認証を設定する方法によっては"共通ファイルノード構成"、クライアント設定の調整が必要になる場合があります。

   a. デフォルトでは、HAクラスタ環境で自動的に接続認証が設定され、が生成されます `connauthfile` に配置/管理されます `<INVENTORY>/files/beegfs/<sysMgmtdHost>_connAuthFile`。デフォルトでは、BeeGFSクライアントの役割は、で定義したクライアントにこのファイルを読み取り/配布するように設定されています `client_inventory.yml`追加のアクションは必要ありません。

      i. 詳細オプションについては、に付属のすべてのデフォルト設定を参照してください "BeeGFSクライアントの役割"。

   b. でカスタムシークレットを指定する場合は、を使用します `beegfs_ha_conn_auth_secret` で指定します `client_inventory.yml` ファイルも同様：

      ```
      beegfs_ha_conn_auth_secret: <SECRET>
      ```

   c. で接続ベースの認証を完全に無効にする場合は、を使用します `beegfs_ha_conn_auth_enabled` で、を指定します `client_inventory.yml` ファイルも同様：

```
beegfs_ha_conn_auth_enabled: false
```

サポートされるパラメータの一覧およびその他の詳細については、を参照してください "BeeGFSクライアントの完全なドキュメント"。クライアントインベントリの完全な例については、をクリックしてください "こちらをご覧ください"。

BeeGFS Client Playbookファイルを作成します

1. 新しいファイルを作成します `client_playbook.yml`

```
# BeeGFS client playbook.
- hosts: beegfs_clients
  any_errors_fatal: true
  gather_facts: true
  collections:
    - netapp_eseries.beegfs
    - netapp_eseries.host
  tasks:
```

2. オプション：NetApp Eシリーズホストコレクションのロールを使用して、クライアントがBeeGFSファイルシステムに接続するためのインターフェイスを設定する場合は、設定するインターフェイスタイプに対応するロールをインポートします。
   a. InfiniBand（IPoIB）を使用している場合は、次の手順を実行します。

```
    - name: Ensure IPoIB is configured
      import_role:
        name: ipoib
```

   b. を使用している環境でRDMA over Converged Ethernet（RoCE）を使用している場合：

```
    - name: Ensure IPoIB is configured
      import_role:
        name: roce
```

   c. 使用しているネットワークがイーサネット（TCPのみ、RDMAはなし）の場合：

```
    - name: Ensure IPoIB is configured
      import_role:
        name: ip
```

3. 最後に、BeeGFSクライアントの役割をインポートしてクライアントソフトウェアをインストールし、フ

ファイルシステムをマウントします。

```
    # REQUIRED: Install the BeeGFS client and mount the BeeGFS file
system.
    - name: Verify the BeeGFS clients are configured.
      import_role:
        name: beegfs_client
```

クライアントのプレイブックの完全な例については、をクリックしてください "こちらをご覧ください"。

BeeGFS Client Playbookを実行します

クライアントをインストール/ビルドしてBeeGFSをマウントするには、次のコマンドを実行します。

```
ansible-playbook -i client_inventory.yml client_playbook.yml
```

BeeGFSの導入を確認します

ファイルシステムを本番環境に導入する前に、ファイルシステムの導入を確認してください。

概要

BeeGFSファイルシステムを本番環境に移行する前に、いくつかの検証チェックを実行します。

手順

1. クライアントにログインして次のコマンドを実行し、想定されるすべてのノードが存在するか到達可能であり、不整合やその他の問題が報告されていないことを確認します。

```
beegfs-fsck --checkfs
```

2. クラスタ全体をシャットダウンし、再起動します。任意のファイルノードから、次のコマンドを実行します。

```
pcs cluster stop --all # Stop the cluster on all file nodes.
pcs cluster start --all  # Start the cluster on all file nodes.
pcs status # Verify all nodes and services are started and no failures
are reported (the command may need to be reran a few times to allow time
for all services to start).
```

3. 各ノードをスタンバイにし、BeeGFSサービスがセカンダリノードにフェイルオーバーできることを確認します。このログインを任意のファイルノードに行うには、次のコマンドを実行します。

```
pcs status # Verify the cluster is healthy at the start.
pcs node standby <FILE NODE HOSTNAME> # Place the node under test in
standby.
pcs status # Verify services are started on a secondary node and no
failures are reported.
pcs node unstandby <FILE NODE HOSTNAME> # Take the node under test out
of standby.
pcs status # Verify the file node is back online and no failures are
reported.
pcs resource relocate run # Move all services back to their preferred
nodes.
pcs status # Verify services have moved back to the preferred node.
```

4. IORやMDTestなどのパフォーマンスベンチマークツールを使用して、ファイルシステムのパフォーマンスが期待どおりであることを確認します。BeeGFSで使用される一般的なテストとパラメータの例については"設計検証"、「BeeGFS on NetApp Verified Architecture」を参照してください。

追加テストは、特定のサイト/設置環境に対して定義された受け入れ基準に基づいて実施する必要があります。

著作権に関する情報

商標に関する情報